

## Завдання

**№ 8**

0	5	10	19	14
10	0	3	11	20
12	15	0	13	3
19	20	6	0	12
19	9	5	12	0

## Розв'язок

Довільний маршрут:  $(1; 2) \rightarrow (2; 3) \rightarrow (3; 4) \rightarrow (4; 5) \rightarrow (5; 1)$

$$F(x) = 5 + 3 + 13 + 12 + 19 = 52$$

	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>d<sub>i</sub></b>
<b>1</b>	$\infty$	5	10	19	14	5
<b>2</b>	10	$\infty$	3	11	20	3
<b>3</b>	12	15	$\infty$	13	3	3
<b>4</b>	19	20	6	$\infty$	12	6
<b>5</b>	19	9	5	12	$\infty$	5

	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>d<sub>i</sub></b>
<b>1</b>	$\infty$	0	5	14	9	5
<b>2</b>	7	$\infty$	0	8	17	3
<b>3</b>	9	12	$\infty$	10	0	3
<b>4</b>	13	14	0	$\infty$	6	6
<b>5</b>	14	4	0	7	$\infty$	5
<b>d<sub>j</sub></b>	7	0	0	7	0	36

$$H = \sum d_i + \sum d_j \quad H_0 = 22 + 14 = 36$$

Редукована матриця:

	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>
<b>1</b>	$\infty$	<b>0</b>	5	7	9
<b>2</b>	0	$\infty$	0	1	17
<b>3</b>	2	12	$\infty$	3	0
<b>4</b>	6	14	0	$\infty$	6
<b>5</b>	7	4	0	0	$\infty$

### Крок 1

	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>d<sub>i</sub></b>
<b>1</b>	$\infty$	<b>0(9)</b>	5	7	9	5
<b>2</b>	0(2)	$\infty$	0(0)	1	17	0
<b>3</b>	2	12	$\infty$	3	0(8)	2
<b>4</b>	6	14	0(6)	$\infty$	6	6
<b>5</b>	7	4	0(0)	0(1)	$\infty$	0
<b>d<sub>j</sub></b>	2	4	0	1	6	

	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>d<sub>i</sub></b>
<b>1</b>	$\infty$	$\infty$	5	7	9	5
<b>2</b>	0	$\infty$	0	1	17	0
<b>3</b>	2	12	$\infty$	3	0	0
<b>4</b>	6	14	0	$\infty$	6	0
<b>5</b>	7	4	0	0	$\infty$	0
<b>d<sub>j</sub></b>	0	4	0	0	0	

$H(1^*; 2^*) = 36 + 9 = \underline{45}$      $H(1; 2) = 36 + 2 = \underline{38} \leq 45$     Включаємо (1; 2).

Виключаємо 5 рядок і 4 стовпець:

	<b>1</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>d<sub>i</sub></b>
<b>2</b>	$\infty$	0	1	17	0
<b>3</b>	2	$\infty$	3	0	0
<b>4</b>	6	0	$\infty$	6	0
<b>5</b>	7	0	0	$\infty$	0
<b>d<sub>j</sub></b>	2	0	0	0	2

### Крок 2

	<b>1</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>d<sub>i</sub></b>
<b>2</b>	$\infty$	0(1)	1	17	1
<b>3</b>	0(4)	$\infty$	3	<b>0(6)</b>	0
<b>4</b>	4	0(4)	$\infty$	6	4
<b>5</b>	5	0(0)	0(1)	$\infty$	0
<b>d<sub>j</sub></b>	4	0	1	6	

$H(3^*; 5^*) = 38 + 6 = \underline{44}$      $H(3; 5) = 38 + 4 = \underline{42} \leq 44$     Включаємо (3; 5).

	<b>1</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>d<sub>i</sub></b>
<b>2</b>	$\infty$	0	1	17	0
<b>3</b>	0	$\infty$	3	$\infty$	0
<b>4</b>	4	0	$\infty$	6	0
<b>5</b>	5	0	0	$\infty$	0
<b>d<sub>j</sub></b>	0	0	0	6	

Виключаємо 3 рядок і 5 стовпець:

	<b>1</b>	<b>3</b>	<b>4</b>	<b>d<sub>i</sub></b>
<b>2</b>	$\infty$	0	1	0
<b>4</b>	4	0	$\infty$	0
<b>5</b>	5	$\infty$	0	0
<b>d<sub>j</sub></b>	4	0	0	4

**Крок 3**

	<b>1</b>	<b>3</b>	<b>4</b>	<b>d<sub>i</sub></b>
<b>2</b>	$\infty$	0(1)	1	1
<b>4</b>	0(1)	0(0)	$\infty$	0
<b>5</b>	1	$\infty$	<b>0(2)</b>	1
<b>d<sub>j</sub></b>	1	0	1	

$H(5^*; 4^*) = 42 + 2 = \underline{44}$      $H(5; 4) = 42 + 0 = \underline{42} \leq 44$     Включаємо (5; 4).

	<b>1</b>	<b>3</b>	<b>4</b>	<b>d<sub>i</sub></b>
<b>2</b>	$\infty$	0	1	0
<b>4</b>	0	0	$\infty$	0
<b>5</b>	1	$\infty$	$\infty$	1
<b>d<sub>j</sub></b>	0	0	1	2

Виключаємо 5 рядок і 4 стовпець:

	<b>1</b>	<b>3</b>	<b>d<sub>i</sub></b>
<b>2</b>	$\infty$	0	0
<b>4</b>	0	0	0
<b>d<sub>j</sub></b>	0	0	0

Відповідно до матриці включаємо в гамільтонів маршрут ребра (2; 3), (4; 1).

$(1; 2) \rightarrow (2; 3) \rightarrow (3; 5) \rightarrow (5; 4) \rightarrow (4; 1)$      $F(x) = 5 + 3 + 3 + 12 + 19 = 42$

## Програмна реалізація

```
#AN INPUT OF DISTANCES

lc = input("How many localities do you have? ");
puts("\nEnter distances:\n");
for i = 1:lc
    inp = input(" ", "s");
    temp = cellfun("str2num", strsplit(inp, " "));
    for j = 1:lc
        distances(i, j) = temp(j);
    endfor
endfor
primary_lc = lc;
primary_distances = distances;

#-----

#REDUCTION

min = inf;
H0 = 0;
min_vect = [];
for k = 1:2
    #transpose matrix
    if (k == 2)
        distances = distances';
    endif
    for i = 1:lc
        for j = 1:lc
            if (distances(i, j) < min)
                min = distances(i, j);
            endif
        endfor
        min_vect(end+1) = min;
        min = inf;
    endfor

    for i = 1:lc
        for j = 1:lc
            distances(i, j) -= min_vect(i);
        endfor
    endfor
    for i = 1:lc
        H0 += min_vect(i);
    endfor
    min_vect = [];
endfor

distances = distances';
```

```

puts("\nReduced matrix:\n");
disp(distances);
printf("\nH0 = %d", H0);

#-----

#ADD EXTRA NUMERATION OF ROWS AND COLS FOR MONITORING
#CHANGES INDEPENDENTLY FROM REDUCTION OF DIMENSIONS

for i = 1:lc
    row_nums(i) = i;
    col_nums(i) = i;
endfor

#-----

#

route = [];
for permanent = 1:100
    printf("\n\n-----\n\nStep %d\n", permanent);

    #REDUCTION CONSTANTS

    min = inf;
    min_vect = [];
    zero_check = 0;
    for k = 1:2
        #transpose matrix
        if (k == 2)
            distances = distances';
        endif
        for i = 1:lc
            for j = 1:lc
                if (distances(i, j) == 0)
                    zero_check++;
                endif
                if (zero_check > 1)
                    min = 0;
                    break;
                elseif (distances(i, j) != 0 && distances(i, j) < min)
                    min = distances(i, j);
                endif
            endfor
            min_vect(k, i) = min;
            min = inf;
            zero_check = 0;
        endfor
    endfor

    distances = distances';

```

```

#ZEROS CALCULATE

max_zero = 0;
temp = 0;
for i = 1:lc
    for j = 1:lc
        if (distances(i, j) == 0)
            temp = min_vect(1, i) + min_vect(2, j);
        endif
        if (temp > max_zero)
            max_zero = temp;
            max_i = i;
            max_j = j;
        endif
    endfor
endfor

#SHOW MATRIX №1
puts("\n1.\n");
disp(distances);

printf("\ndi =");
disp(min_vect(1, :));
printf("\ndj =");
disp(min_vect(2, :));

#infinite to node in next table at coords (i,j)
distances(max_i, max_j) = inf;

#SHOW MATRIX №2
puts("\n2.\n");
disp(distances);

#infinite to node in next table at reverse coords (j,i)
distances(max_j, max_i) = inf;

#DELETE i ROW & j COLUMN & INDEXES FIX
distances(max_i, :) = [];
distances(:, max_j) = [];

fcoord_node = row_nums(max_i);
scoord_node = col_nums(max_j);
row_nums(max_i) = [];
col_nums(max_j) = [];
lc--;

#LAST REDUCTION
sum_for_H2 = 0;

```

```

min = inf;
#min_vect = [];
for k = 1:2
    #transpose matrix
    if (k == 2)
        distances = distances';
    endif
    for i = 1:lc
        for j = 1:lc
            if (distances(i, j) < min)
                min = distances(i, j);
            endif
        endfor
        #min_vect(k, i) = min;
        if (min > 0)
            for j = 1:lc
                distances(i, j) -= min;

            endfor
            sum_for_H2 += min;
        endif
        min = inf;
    endfor
endfor
distances = distances';

#SHOW MATRIX №3
puts("\n3.\n");
disp(distances);

#CHECK INCLUDE
H1 = H0 + max_zero;
H2 = H0 + sum_for_H2;

printf("\nH(%d*; %d*) = %d + %d = %d", max_i, max_j, H0, max_zero, H1);

if (H2 <= H1)
    printf("\nH(%d; %d) = %d + %d = %d <= %d\n", max_i, max_j, H0,
sum_for_H2, H2, H1);
    H0 = H2;
    #ADD NODE TO ROUTE
    route(end+1, 1) = fcoord_node;
    route(end, 2) = scoord_node;
endif

#CHECK STOP

check_stop = 0;
for i = 1:lc

```

```

        for j = 1:lc
            if (distances(i, j) == 0 || distances(i, j) == inf)
                check_stop++;
            endif
        endfor
    endfor
    if (check_stop == lc*lc)
        break;
    endif
    check_stop = 0;

endfor

#-----

#FINDING LAST TWO ROUTES

#last-1 route
for i = 1:lc
    if (isinf(distances(1, i)) == 0)
        route(end+1, 1) = row_nums(1);
        route(end, 2) = col_nums(i);
    endif
endfor

#last route

for i = 1:lc
    if (col_nums(i) == route(1, 1))
        route(end+1, 1) = row_nums(2);
        route(end, 2) = col_nums(i);
    endif
endfor

#SORTING ROUTES

sorted_routes = [];
sorted_routes(end+1, 1) = route(1, 1);
sorted_routes(end, 2) = route(1, 2);
route(1, :) = [];
check_fullsorted = 1;
while (check_fullsorted < primary_lc)
    for i = 1:primary_lc-check_fullsorted
        if (sorted_routes(end, 2) == route(i, 1))
            sorted_routes(end+1, 1) = route(i, 1);
            sorted_routes(end, 2) = route(i, 2);
            route(i, :) = [];
            check_fullsorted++;
        break;
    endif
endfor

```



**endwhile**

#DISPLAY ROUTE

puts("\nRoute: ");

**for** i = 1:primary\_lc

**if** (i == primary\_lc)

        printf("(%d,%d)\n", sorted\_routes(i,1), sorted\_routes(i,2));

**else**

        printf("(%d,%d)->", sorted\_routes(i,1), sorted\_routes(i,2));

**endif**

**endfor**

F = 0;

**for** i = 1:primary\_lc

    F += primary\_distances(sorted\_routes(i,1), sorted\_routes(i,2));

**endfor**

printf("F = %d\n", F);

## Результат роботи програми

Reduced matrix:

Inf	0	5	7	9
0	Inf	0	1	17
2	12	Inf	3	0
6	14	0	Inf	6
7	4	0	0	Inf

H0 = 36

-----

Step 1

1.

Inf	0	5	7	9
0	Inf	0	1	17
2	12	Inf	3	0
6	14	0	Inf	6
7	4	0	0	Inf

di =    5    0    2    6    0

dj =    2    4    0    1    6

2.

Inf	Inf	5	7	9
0	Inf	0	1	17
2	12	Inf	3	0
6	14	0	Inf	6
7	4	0	0	Inf

3.

Inf	0	1	17
0	Inf	3	0
4	0	Inf	6
5	0	0	Inf

H(1\*; 2\*) = 36 + 9 = 45

H(1; 2) = 36 + 2 = 38 <= 45

-----

Step 2

1.

Inf	0	1	17
0	Inf	3	0
4	0	Inf	6
5	0	0	Inf

di = 1 0 4 0

dj = 4 0 1 6

2.

Inf	0	1	17
0	Inf	3	Inf
4	0	Inf	6
5	0	0	Inf

3.

Inf	0	1
0	0	Inf
1	Inf	0

$H(2^*; 4^*) = 38 + 6 = 44$

$H(2; 4) = 38 + 4 = 42 \leq 44$

-----

Step 3

1.

Inf	0	1
0	0	Inf
1	Inf	0

di = 1 0 1

dj = 1 0 1

2.

Inf	0	1
0	0	Inf
1	Inf	Inf

3.

Inf	0
0	0

$H(3^*; 3^*) = 42 + 2 = 44$

$H(3; 3) = 42 + 0 = 42 \leq 44$

Route: (1,2) -> (2,3) -> (3,5) -> (5,4) -> (4,1)

F = 42